Project HomeFinder

1st Ilian Sebti
Department of Engineering
Hanyang University
Seoul, Korea
Group HomeFinder
ilian.sebti@edu.devinci.fr

2nd William Olsson
Department of Engineering
Hanyang University
Seoul, Korea
Group HomeFinder
lobbeolsson@gmail.com

3th Jeong JinYeong
Department of Engineering
Hanyang University
Seoul, Korea
Group HomeFinder
tttjjjyyy1234@daum.net

4th Enkhjin Puntsagnorov
Department of Information systems
Hanyang University
Seoul, Korea
Group HomeFinder
enhjinnasaa48@gmail.com

5rd Nina Lauria
Department of Information systems
Hanyang University
Seoul, Korea
Group HomeFinder
ninalauria01@gmail.com

***Abstract*—For our project, we would like to develop a new website that shifts the focus of property search away from geographical features and onto the immediate surroundings of a potential new home. The purpose of this project is to refine and tailor the apartment search experience to the users' needs. Users should be able to filter their search based on self-selected criteria to obtain the best results for them. The website should have the capability to find all available listings that meet the user's chosen filters. The results will be displayed on a map, and users can save these results to their favorites for future reference. The advantage of such a search is that users won't have to conduct their own research to discover what's nearby the apartment. With the map-based search, it's easy for users to see the Points of Interest in the vicinity of the location. With enough input data, the website can suggest the best home for the user, making the apartment search more enjoyable and time-saving.**

I. Role Assignments

Roles	Name	Task description and etc.
User	Nina Lauria	Users navigate through the platform, employing various filters to refine their home searches, while also utilizing functionalities like sorting and favoriting to enhance their experience. They may create accounts for personalized interactions and seek assistance through help sections when needed.
Customer	Ilian Sebti	Customers, such as real estate agencies or sellers, manage their property listings on the platform, ensuring accurate and appealing presentation. They utilize provided analytics and insights to understand market trends and user behavior, adjusting their strategies to enhance visibility and engagement accordingly.
Software developer	Jeong JinYeong / Enkhjin Puntsagnorov	Software developers engage in coding, testing, and optimizing the platform, ensuring its functionalities from search features to data security are robust and user-friendly. They collaborate with stakeholders to align technological implementations with the platforms vision, ensuring scalability and adaptability to evolving technologies.
Development manager	William Olsson	Development managers oversee the platform's development process, ensuring alignment with quality and requirements. They manage task distribution among the software development team, facilitate resolution of challenges, and ensure the project adheres to timelines and budgets, while maintaining a focus on quality and user experience.

II. Introduction

A. Motivation

Our motivation initially sprung from our personal struggles and those of others around us. The collective frustration regarding the conventional online house hunting experience, the endless tabs, the disjointed correlation between home and locality, and the exhaustive manual research struck a chord with us. We realized that while physical properties of a home (such as size, number of rooms, etc.) are pivotal, the impact of location features on the living experience was a significantly underrepresented factor in existing platforms.

We believe that a home is not just a physical space but an environment that significantly influences one's quality of life. Recognizing the paramount importance of convenience,

accessibility, and connectivity in modern living, we were driven to devise a solution that prioritizes these aspects. Our platform is not just a tool; it's a companion that assists users in finding a home that resonates with their lifestyle, needs, and preferences.

We were also motivated by the prospect of empowering users to make well-informed decisions. By presenting a visual, map-based experience, we offer users a comprehensive view that enables them to evaluate and compare homes based not just on the properties but also on the surrounding infrastructures, such as transportation links, grocery stores, schools, and hospitals. This integrative approach ensures that users can visualize their life in a potential home, foreseeing the conveniences and challenges that the location presents.

B. Problem Statement

The process for creating the problem statement involved several steps. Initially, we identified the problem area of apartment hunting, focusing on the issue that location features are often not adequately considered when searching for a new home. Next, we analyzed the causes and effects of the problem and developed a potential solution.

Following that, we conducted a user forecast to determine our primary target audience and identify other potential beneficiaries of our product. The user forecast also aimed to validate whether there is indeed a demand for such a product.

Online searching for a new abode generally bases itself on the physical properties of the house or apartment. Prioritizing these properties over location features results in an incomplete picture and search experience. Even though the search results are displayed on a map, you still have to research manually to find out where the necessary things are located near your potential new home.

For this reason, we create a platform that turns the search for a new place to live into a visual, map-based experience. It is easy to see which Points of Interest are located near the places you are looking at, and you also see how to get there. Moreover, with sufficiently input data, we could actually suggest the best home for the user.

The solution is suitable for users who do not have to live in a specific place but rely on the availability and connection (transportation) around the property. Companies can use the aggregated data from the real estate platform to understand needs and incorporate the insights into real estate development and real estate pricing.

III. Related Software

A. Zillow

Zillow is an online real estate marketplace that provides a plethora of data related to available properties, including price estimates, aerial views, and comparative market analyses.

While Zillow does present a wide array of property options with basic location mapping, our platform aims to go a step further by intensively integrating location features into the search experience. Zillow tends to prioritize physical property features and pricing data, whereas our platform is envisaged to offer a visual, map-based searching experience that prominently highlights Points of Interest (POIs) and demonstrates how to navigate to them.

B. Zigbang

Zigbang stands as the premier real estate Service in South Korea, boasting over 30 million users and a wealth of user data. Not only does it sustain a dominant market presence, but it is also dynamically expanding its offerings by introducing an array of services such as 'IoT service' and 'VR Home Tour service', complementing their staple 'on-tact real estate sales service'. Zigbang notably runs a 'real estate recommendation service' which closely mirrors our 'HomeFinder', characterized by:

- Recommending properties using parameters like school districts, transportation accessibility, and selling prices.
- Enabling users to stay abreast of new listings in real-time via the Favorites feature.

Despite its strengths, Zigbang's recommendations hinge on a limited set of criteria, such as school district and selling price. In contrast, HomeFinder enhances the personalization of its service, recommending properties based on an expanded range of criteria that also encompasses nearby gyms, cafes, and convenience stores.

C. Dabang

Dabang, the second-leading real estate Service in South Korea, has secured 20 million users, providing it with a substantial data reservoir, albeit not as extensive as Zigbang's. Dabang excels in customization within its fundamental service, "on-tact real estate sales service", standing distinctively in comparison with Zigbang by recommending properties based on a multifaceted set of criteria: selling price, property type (rental, lease, sales), square footage, number of floors, parking availability, and elevator availability, among others.

Nonetheless, the service is not without its shortcomings, chiefly that its criteria predominantly spotlight the property's internal characteristics. While the internal facets of a residence are undeniably significant, external factors like proximity to convenience stores and gyms also wield importance in enhancing life quality. Hence, HomeFinder aims to uplift user satisfaction by recommending properties that judiciously consider both internal and external conditions.

D. Naver Real Estate

Naver Real Estate, being the third largest real estate service in South Korea, carries its own set of strengths concerning capital, brand recognition, and a substantial user pool, owing to its inception by Naver, a prominent IT conglomerate. Unlike its counterparts, this service prioritizes information dissemination over direct transactions. Leveraging the robust platform of Naver, it dominates in real estate listings, claiming a remarkable 95 percent share.

Nevertheless, like Zigbang and Dabang, Naver Real Estate concentrates solely on providing a filtering function for the internal conditions of a property and makes uploading photos to

visualize property conditions a challenging task. HomeFinder, therefore, aspires to boost user satisfaction by recommending properties with a balanced consideration of both internal and external conditions, and facilitating the provision of photos to assess the property's condition.

IV. Requirements

The information architecture of the website is well-structured and logically constructed. The main categories of the website are the homepage, the resultpage, the detail page, and the contact page.

A. Homepage

- The website must be capable of filtering user searches based on selected criteria, such as location, price range, property type, number of rooms, etc.
- Users should be able to select as many search filters as they wish to refine their search results according to their preferences and needs.
- Intuitive and interactive user interface components should be employed to facilitate easy navigation and usage of search filters.

B. Result Page

- Results should be displayed along with a dynamic map to provide users with a clear spatial understanding of the property locations relative to various Points of Interest.
- The map should be interactive, allowing users to zoom, pan, and click on property markers for more detailed information.
- Users should have the ability to sort search results based on various parameters, such as price (ascending and descending), proximity to Points of Interest, property size, etc.
- The sorting functionality should be straightforward, enabling users to quickly rearrange their search results according to the selected parameter.

C. Favorites Page

- Users should be able to add listings to their favorites for easy access and future reference.
- The favorites page should provide a simplified view of the selected properties, with options to visit the detailed view or remove items from the favorites list.
- It would be beneficial to allow users to categorize or label their favorites for enhanced organization and retrieval.

D. Account Creation Page

- Users should be able to create an account using a simple and secure sign-up process, with necessary validations to ensure data accuracy and security.
- Account management options should be provided to allow users to update their personal information, preferences, and password.

E. View of Nearby Services

- "Nearby Services" provides users with detailed information about essential amenities, such as grocery stores, schools, healthcare facilities, and public transportation options in the vicinity of rental properties.
- Users can access comprehensive details about these services, including their operating hours.

F. Rental term

- User can filter the duration of rent. So that available apartment or house will be shown as time by time. This filter allowing users to select the duration of the lease they are looking for.

G. Accessibility

- User should be able to see the accessibility of apartment.
- Accessibility filter provide a quick view of stairs and elevator information. For each apartment list, collect and store information about the number of floors and the presence or absence of elevators in that apartment.

V. Development environment

A. Choice of software development platform

In orchestrating our development environment, we have strategically selected each technology to optimize our workflow for peak performance, flexibility, and efficiency. **For the frontend, React.js emerges as our choice** for its state-of-the-art capabilities in crafting dynamic user interfaces that engage and respond with agility. Our commitment to a seamless user experience is further evidenced by the integration of **HTML and CSS**, which serve as the bedrock for a solid and aesthetically pleasing design framework.

The backend architecture is entrusted to the robust Node.js, coupled with Express.js for its lightweight and flexible framework that accelerates our backend development. We strategically incorporate **Java** for certain backend modules, capitalizing on its storied reliability and inherent security features, which are paramount for a cross-platform enterprise application.

Data management is the backbone of any modern web application, and MongoDB stands as our database of choice. Its agile NoSQL structure allows us to adeptly handle and scale with the large, unstructured datasets that our application generates and consumes.

Development tools are the artisans' instruments, and **Visual Studio Code** is our chosen palette, offering unmatched support for a wide array of programming languages and extensions that empower our developers to write, debug, and deploy with confidence. Essential to our process are tools like **TODO Tree and CodeSee,** which bring clarity and organization to our codebase, and **Insomnia**, which serves as our gateway for thorough API testing and debugging. The collaborative nature of our project is facilitated by **GitHub**, a cornerstone for our version control systems, ensuring that our code evolves in harmony with multiple contributors.

A robust development environment is more than software; it's also the hardware that supports it. Hence, we mandate a minimum of 8 GB of RAM for our systems to ensure that our suite of development tools operates at full stride without bottlenecks, allowing our team to maintain a smooth and efficient development cadence.

B. Cost estimated

Given the nature of our project, the focus is on cost-efficiency; therefore, we aim to utilize free resources wherever possible. This commitment extends to our development environment, where we are using Visual Studio Code—an open-source code editor—across all operating systems.

C. Software in Use

Our development toolkit comprises specific, up-to-date versions of software essential for our project's success:

- Employing React.js version 17 or later, we ensure that our front-end architecture is both contemporary and performant.
- With Node.js version 16 or above, our back-end is scalable and responsive to real-time data flow.
- Express.js complements Node.js, offering a streamlined framework for our web application's infrastructure.
- MongoDB, a NoSQL database, is our choice for efficient data storage and retrieval, pivotal for handling our application's data demands.
- Visual Studio Code serves as the cornerstone of our development, with its extensibility and diverse language support, including vital extensions for productivity.
- Insomnia is integral to our API lifecycle, ensuring that each endpoint is robustly tested and debugged.
- GitHub underpins our collaborative efforts, providing a robust platform for code versioning and team collaboration.

The operating systems supporting our development include Windows 11, macOS, and Linux. We require a minimum of 8 GB of RAM for our computer systems to ensure efficient multitasking and smooth operation of our development tools.

D. Task distribution

Name	Roles	Task description
Nina Lauria	Documentation and Frontend Leader	Frontend tasks and responsible for UI testing and feedback. Also oversees the documentation to ensure it's updated and does frontend tasks.
Jeong JinYeong / Ilian Sebti	Backend DevOps	Backend tasks like setting up the server, database integration, API design and implementation.
William Olsson	Project and Backend leader	Integration of Frontend and Backend, does also DevOps tasks in both.
Enkhjin Puntsagnorov	Frontend DevOps	Is tasked with the development and optimization of the frontend, ensuring the user interface is both functional and user-friendly.

VI. Specifications

The required functionalities of our project are divided between multiple web pages.

A. Homepage / Search page

Through this page, users will be able to search for a home by using a filter based search engine. The selectable criterias for the filters will be mostly property oriented (City, Price, number of rooms, energy efficiency...). Users should be able to select multiple filters to refine their search results. We did clickable Mockups for better visualization. To see the Mockups click here.

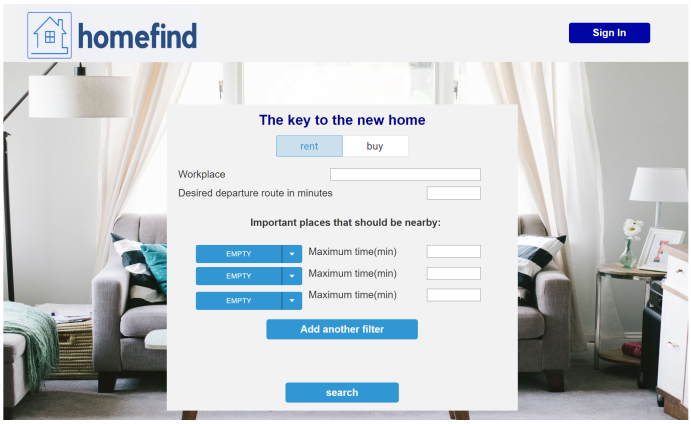

Fig. 1: Prototype of the Homepage / Search page

Backend Implementation:

- Define API endpoints for property search.
- Implement filter-based search logic, which includes receiving filter parameters and querying the database for matching properties.
- Return the search results in JSON format to the frontend.

Frontend Implementation:

- Create the search form with filter options (City, Price, Number of Rooms, etc.).
- Capture user input and make an API request to the backend.
- Display the search results on the frontend, possibly as a list of properties that meet the criteria.

B. Result Page

This page will display the results of the search on a dynamic map, giving the user the ability to rearrange the search results and give a clear spatial understanding of the property locations relative to various Points of Interest. From this page, users will be able to add properties to their favorites. Backend Implementation:

- Create an API endpoint to fetch the list of properties for the result page.
- Implement logic to retrieve property details, including geolocation information.
- Return the property details in JSON format to the frontend.

Frontend Implementation:

- Display the search results on a dynamic map, using JavaScript mapping libraries like Mapbox or Google Maps.
- Allow users to rearrange search results on the map.
- Implement the option to add properties to the user's favorites list, sending requests to the backend to update the user's favorite properties.

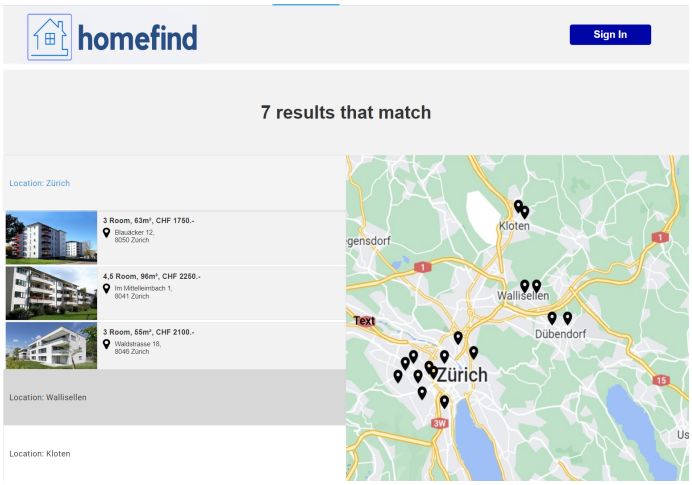

Fig. 2: Prototype of the Result Page

C. Favorites

From this page, users will be able to view a listing of their favorite properties as well as a simplified view of them with the option to access the detailed view or to remove the property from the list. Frontend Implementation:

- Create a user interface for viewing a listing of favorite properties.
- Implement options for users to access the detailed view of properties or remove them from the list.
- Send requests to the backend to manage the user's favorite properties.

D. Account creation/login

Backend Implementation:

- Implement user authentication and authorization mechanisms, using libraries like Flask-Login or Django's authentication system.
- Create API endpoints for user registration, login, and account management.

Frontend Implementation:

- Create user registration and login forms.
- Capture user input, make API requests to the backend for registration and login, and manage user sessions.
- Implement UI elements for user account management, including a user profile page.

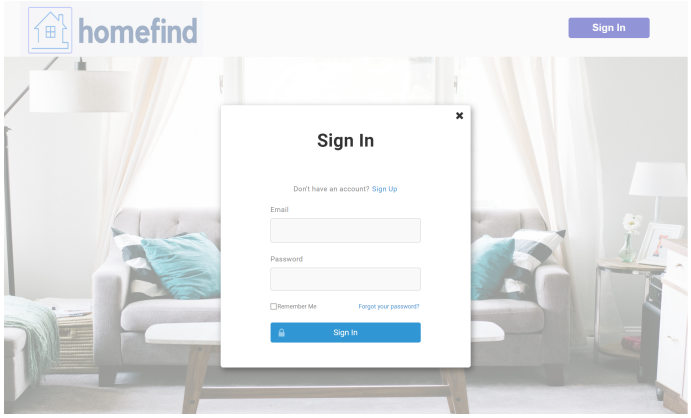

Fig. 3: Prototype of the Account creation/login

E. View of nearby Services

Use third-party APIs (e.g., Google Places API) to retrieve data on amenities like grocery stores, schools, healthcare facilities, and public transport and design database schema to store information of amenities linked to property locations.

Backend Implementation:

- Develop an endpoint that takes property location as input and queries the table for nearby services.

Frontend Implementation:

- Implement a map interface (using libraries like Leaflet.js or Google Maps JavaScript API) that displays nearby services as markers.
- Create a UI component that shows a detailed list of amenities when a user selects a property.

F. Rental Term

The rental term feature requires a UI component that allows users to define the duration of the lease they are interested in. This could take the form of a slider or a set of dropdown options. The property search functionality would be enhanced to interpret the rental term as a search parameter, filtering the database results accordingly.

Our database schema would be structured to include a 'rentalTerm' field within the properties table, which would reflect the available durations of lease for each listing. Automated tests would be crucial to ensure the filter operates as intended, only showing users the properties available for their specified term.

G. Accessibility

Accessibility information is pivotal for many users. To accommodate this, property input forms would be updated to capture details regarding elevators and the number of floors. The database schema would evolve to include an 'accessibilityFeatures' table, which would store this information linked to each property.

We would develop backend logic to provide this information through an API endpoint, which would be called when a user views property details. The frontend would display this information clearly, using universally recognized icons or labels to indicate accessibility features.